International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075, Volume-3, Issue-1, October 2013

Parallel CRC Generation for High Speed Applications

B. Naveen, K. Swaraja, M. C. P Jagdissh

Abstract *-- Cyclic redundancy check is commonly used in data communication and other fields such as data storage and data compression, as a essential method for dealing with data errors. Usually, the hardware implementation of CRC computations is based on the linear feedback shift registers (LFSRs), which handle the data in a serial way only, Though the serial calculation of the CRC codes cannot achieve a high throughput. parallel CRC calculation can significantly increase the throughput of CRC computations. Variants of CRCs are used in applications like CRC-16 BISYNC protocols, CRC32 bit in Ethernet frame for error detection, CRC8 bit in ATM, CRC-CCITT in X-25 protocol, disc storage, SDLC, and XMODEM.*

High speed data transmission is the current scenario in networking environment. Cyclic redundancy check (CRC) is essential method for detecting error when the data is transmitted. About the speed of transmitting data, and to synchronize with speed, it is necessary to increase speed of CRC generation. Starting from the serial architecture a recursive formula was used from which parallel design is obtained. But in this paper presents 64 bits parallel CRC architecture based on F matrix with order of generator polynomial is 32. It is hardware efficient and required 50% less cycles to generate CRC with same order of generator polynomial.

In this architecture w= 64 (input) bits are parallel processed and order of generator polynomial is m= 32. If 32 bits are processed parallely then CRC-32 will be generated after (k +m)/w cycles.Where 'k' indicates number of data bit and 'm' indicates the order of generator polynomial If we increase number of bits to be processed parallely, number of cycles required to calculate CRC can be reduced.

Keywords- Cyclic Redundancy Check, Parallel CRC calculation, Linear Feedback Shift Register, LFSR, F matrix.

I. INTRODUCTION

Digital communication system is used to transport an information bearing signal from the source to a user destination via a communication channel.

Cyclic redundancy check is commonly used in data communication and other fields such as data storage and data compression, as a essential method for dealing with data errors [6]. Usually, the hardware implementation of CRC computations is based on the linear feedback shift registers (LFSRs), which handle the data in a serial way. the serial calculation of the CRC codes cannot achieve a high throughput. In contrast, parallel CRC calculation can significantly increase the throughput of CRC computations. here the throughput of the 32-bit parallel calculation of CRC-32 can achieve several gigabits per second [1.]

Manuscript Received on October 2013.
B. Naveen, ECE dept, MREC, Secunderabad, India.
K. Swaraja, ECE dept., MREC, Secunderabad, India.
Dr. M. C. P Jagdissh, HOD, ECE dept. MREC, Secunderabad, India.

However, that is still not enough for high speed application in Ethernet networks. A possible solution is to send more bits in parallely Variants of CRCs are used in applications like CRC-16 BISYNC protocols, crc32 in Ethernet frame for error detection. CRC8 bit is used in ATM, CRC-CCITT used in X-25 protocol and disc storage, SDLC, and XMODEM. Albertengo and Sisto [2], has proposed z transform based architecture for parallel CRC, which it's not possible to write synthesizable VHDL code. Braun et al [4] presented an approach suitable for FPGA implementation; FPGA has very complex analytical proof. Another approach based on Galois field has been proposed by Shiehetal. [3]. Campobello [1], has presented pre-calculated F matrix based 32 bit parallel processing, here pre-calculation doesn't work when the polynomial is change. So. the proposed architecture deal with 64bit parallel processing based on built in F matrix generation; this gives CRC with half number of cycles. This paper starts with the introduction of serial CRC generation based on LFSR. F matrix based parallel architecture for 32 bits and 64 bits are described in section 3 , 4. Finally, simulation output results are shown in section 5 and concluded in section 6.

II. MATRIX GENERATION

$$F = \begin{bmatrix} P_{m-1} & 1 & 0 & 0 & 0 \\ P_{m-2} & 0 & 1 & 0 & 0 \\ P_{m-3} & 0 & 0 & 1 & 0 \\ P_{m-4} & 0 & 0 & 0 & 1 \\ .. & .. & .. & .. & .. \\ P_0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Where,{p0……pm-1}is generator polynomial. For example, the generator polynomial for CRC4 is {1, 0, 0, 1, 1} and w bits are parallel processed.

$$F = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Here w=m=4, for that F^w matrix calculated as follow.

$$F^4 = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Parallel architecture based on F matrix As shown in fig. d is data that is parallel processed (i.e 32bit), X' is next state, X is current state (generated CRC), F(i)(j) is the i^{th} row and j^{th} column of F^w matrix.

If X = [x_{m-1} …..x_1 x_0]T is utilized to denote the state of the shift registers, in linear system theory, the state equation for LFSRs can be expressed in modular 2 arithmetic as follow.

$$X_i' = (P0 \otimes X_{m-1}) \otimes X_{i-1}$$

Where, X(i) represents the i^{th} state of the registers, X(i + 1) denotes the $(i + 1)^{th}$ state of the registers,here d means one bit shift-in serial input. F is an m x m matrix and G is a 1 x m matrix.

$$G = (00……..01)T$$

Furthermore, if F and G are substituted by Equations (4) and (5), we can rewrite equation in the matrix form as:

$$\begin{bmatrix} X'_{m-1} \\ X'_{m-2} \\ . \\ X'_0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} X_{m-1} \\ X_{m-2} \\ . \\ X_0 \end{bmatrix} \oplus \begin{bmatrix} 0 \\ 0 \\ . \\ 1 \end{bmatrix} .d$$

Finally, equation (6) can be written in matrix form as

$$X' = F^W \otimes X \otimes D$$

Equation is illustrated in fig. If w bits are parallel processed, then CRC will be generated after (k +m)/w Equation (8) can be expanded for CRC4 given below.

$$X_3' = X_2 \otimes X_1 \otimes X_0 \otimes D_3$$

$$X_2' = X_3 \otimes X_2 \otimes D_2$$

$$X_3' = X_3 \otimes X_2 \otimes X_1 \otimes D_1$$

$$X_0' = X_3 \otimes X_2 \otimes X_1 \otimes X_0 \, D_0$$

III. CRC 32 BIT (EXISTING TECHNOLOGY)

➢ *F matrix based parallel CRC generation*

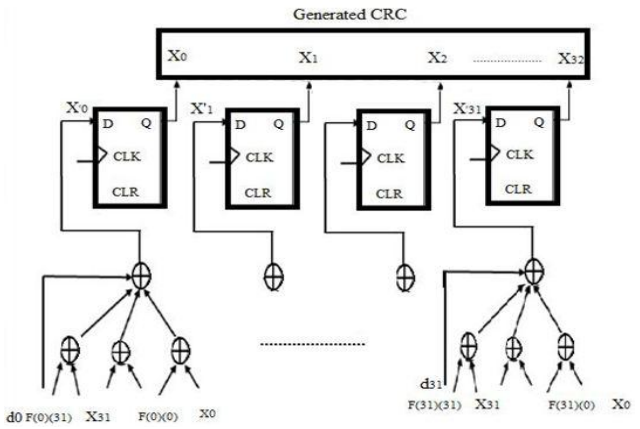

Figure 1. Parallel calculation of CRC-32 for 32bit

A parallel implementation of the CRC can be derived from the above considerations. Yet again, it consists one special register. this case the inputs of the FFs are the exclusive sum of some FF outputs and inputs. Fig shows a possible implementation. More precisely is equal to the value at the i^{th} row and j^{th} column. Even in the case of, if the divisor is fixed, then the AND gates are unnecessary.

Furthermore, the number of FFs remains unchanged. We recall that if then inputs are not needed. Inputs are the bits of dividend sent in groups of bits each. As to the realization of the LFSR2, by considering we have a circuit very similar to that inputs are XORed with FF outputs and results are fed back.

Property of the F^w matrix and the previously mentioned fact that can be regarded as a recursive calculation of the next state X' by matrix F^w, current state X and parallel input D, make the 32-bit parallel input vector suitable for any length of messages besides the multiple of 32 bits. Remember that the length of the message is byte based. If the length of message is not the multiple of 32,after a sequence of 32-bit parallel calculation, the final remaining number of bits of the message could be 8; 16.

For all these situations, an additional parallel calculation w = 8; 16; 24 is needed by choosing the corresponding F^w. Since F^wcan be easily derived from F32, the calculation can be performed using Equation within the same circuit as 32- bit parallel calculation, the only difference is the F^w matrix. If the length of the message is not the multiple of the number of parallel processing bits w = 4 i.e. data bit is11011101011.

Then last two more bits (D (3)) need to be calculated after getting X (12). Therefore, F2 must be obtained from matrix F4, and the extra two bits are stored at the lower significant bits of the input vector D. Equation can then be applied to calculate the final state X (14), which is the CRC code.

Therefore, only an extra cycle is needed for calculating the extra bits if the data message length is not the multiple of w, the number of parallel processing bits. It is worth to notice that in CRC-32 algorithm, the initial state of the shift registers is preset to all `1's. Therefore, X(0) = 0xFFFF. However, the initial state X (0) does not affect the correctness of the design. In order for better understanding, the initial state X (0) is still set to 0x0000 when the circuit is implemented.

IV. CRC 64 BIT (PROPOSED TECHNOLOGY)

In proposed architecture w= 64 bits are parallel processed and order of generator polynomial is m= 32 as shown in fig.

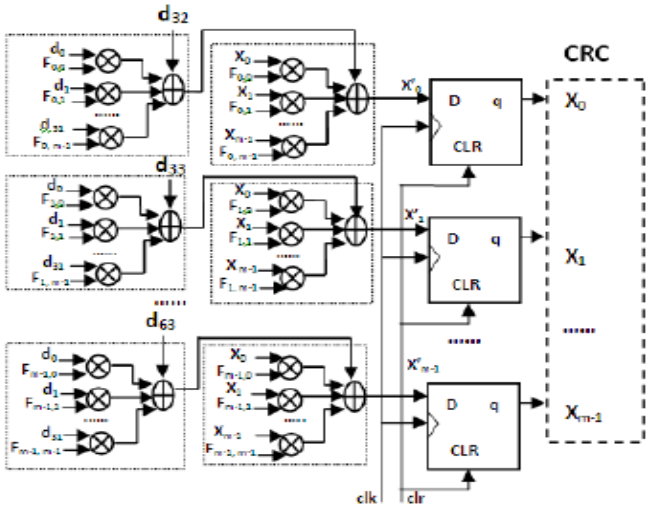

Figure 2. 64-bit parallel calculation of CRC-32

As discussed in section 3, if 32 bits are processed parallel then CRC-32 will be generated after (k +m)/w cycles. If we increase number of bits to be processed parallel, number of cycles required to calculate CRC can be reduced. Proposed architecture can be realized by below equation.

$$Xtemp = F^{W} \otimes D(0to31) \oplus D(32to63)$$

$$X' = F^{W} \otimes X \oplus Xtemp$$

Where,

D (0 to 31) =first 32 bits of parallel data input

D (0 to 63) = next 32 bits of parallel data input

X'=next state X=present state

In proposed architecture d*i* is the parallel input and F(i)(j) is the element of F32 matrix located at i^{th} row and j^{th} column.

As shown in figure 3 input data bits d0….d31 anded with each row of F^{W} matrix and result will be xored individually with d32, d33 …….d63. Then each xored result is then xored with the X' (i) term of CRC32. Finally X will be the CRC generated after (k +m)/w cycle, where w=64.

V. RESULT AND ANALYSIS

The proposed architecture is synthesized in Xilinx-9.2i and simulated in Xilinx ISE Simulator, which required half cycle then the previous 32bit design[1][5]. In our programming in VHDL by specifying only generator polynomial, it directly gives F matrix useful for parallel CRC generation that is not available in previous methods [1][5][6]. Hardware utilization is compared in table I for different approaches for different parameter like LUT, CPD and cycle.

Comparison Of Lut's And Clock Cycle's

CRC w parallel bits	Clock cycles	LUTS	CPD
CRC 32 W=32 bit[1]	17	162	7.3ns
CRC 32 W=32 bit[8]	17	220	30.5ns
CRC 32 W=64 bit[proposed]	9	331	5,7ns

From the table it is observe that, the architecture proposed by [1][8] require 17 clock cycle to generate CRC as per equation (k+ m)/w and for proposed architecture it required only 9 cycle, CPD for proposed architecture is less than the architecture [1][8],only the disadvantage for proposed architecture is the no. of LUT get increased, so area also get increase .

VI. CONCLUSION

32bit parallel architecture required 17 ((k + m)/w) clock cycles for 64 byte data. Proposed design (64bit) required only 9 cycles to generate CRC with same order of generator polynomial. So, it drastically reduces computation time to 50% and same time increases the speed. Pre calculation of F matrix is not required in proposed architecture.

ACKNOWLEDGMENT

I would like to thank our guide **Mrs. K. Swaraja,** Associate Professor**,** Department of ECE for his valuable support in various ways.

I express my sincere thanks to our **Dr. M. Ch. P. Jagdissh,** Head of ECE Department for helping me in carrying out this project.

REFERENCES

[1] Campobello, G.; Patane, G.; Russo, M.; "Parallel CRC realization," *Computers, IEEE Transactions on* , vol.52, no.10, pp. 1312- 1319, Oct.2003

[2] Albertengo, G.; Sisto, R.; , "Parallel CRC generation," *Micro,IEEE* , vol.10, no.5, pp.63-71,Oct1990

[3] M.D.Shieh et al., "A Systematic Approach for Parallel CRC Computations," Journal of Information Science and Engineering, May 2001.

[4] Braun, F.; Waldvogel, M.; , "Fast incremental CRC updates for IP over ATM networks," *High Performance Switching and Routing,2001 IEEE Workshop on* , vol., no., pp.48-52, 2001

[5] Weidong Lu and Stephan Wong, "A Fast CRC Update Implementation", IEEE Workshop on High Performance Switching and Routing ,pp. 113-120, Oct. 2003.

[6] S.R. Ruckmani, P. Anbalagan, " High Speed cyclic Redundancy Check for USB" Reasearch Scholar, Department of Electrical Engineering, Coimbatore Institute of Technology, Coimbatore-641014, DSP Journal, Volume 6, Issue 1, September, 2006.

[7] Yan Sun; Min Sik Kim; , "A Pipelined CRC Calculation Using Lookup Tables," *Consumer Communications and Networking Conference (CCNC), 2010 7th IEEE* , vol., no., pp.1-2, 9-12 Jan. 2010

[8] Sprachmann, M.; , "Automatic generation of parallel CRC circuits," *Design & Test of Computers, IEEE* , vol.18, no.3, pp.108-114, May 2001.

Authors Profile

B. Naveen
is presently pursuing final semester M.Tech in Digital Systems & Computer Electronics at Mallareddy Engineering college, Secunderabad. He received degree B.E in Electronics and communication From KCEA. His areas of interest are Signal Processing,VLSI, DSP algorithm And Architecture.

K. Swaraja
is presently working as a Associate Professor in the department of Electronics and Communication Engineering, MREC, Secunderabad, Andhra Pradesh, India. She is having 12 years of teaching experience. Her areas of interest are Communication systems, Image Processing. Digital signal processing.

Dr. M. J. C. Prasad
is presently working as a Head of the department of Electronics and Communication Engineering, MREC, Secunderabad, Andhra Pradesh, India. He is having 15 years of teaching experience. His areas of interest are Communication systems, Digital Systems, Image Processing. Digital signal processing, Advance DSP Systems.